Remarks

Applicants respectfully request reconsideration of the present application in view of the foregoing amendments and the following remarks. Claims 41-87 are pending in the application. No claims have been allowed. Claims 58-60 have been amended. Claims 41-50, 63-71 and 75-87 are withdrawn from consideration. Claims 51-62 and 72-74 are rejected. These rejections are respectfully traversed. Claims 41, 51, 58, 63, 69, 72, 75, 79, and 84 are independent.

Patentability of Claim 58 under 35 U.S.C. 101

The Action rejects claim 58 under 35 U.S.C. 101, asserting that the claimed invention is directed to non-statutory subject matter. Specifically, the Action asserts that claim 58 does not "show any functional descriptive materials for getting the utility output, therefore they are missing the patentable utilities" and that the "claims appear to use functions or definitions without providing a useful concrete and tangible result."

Claim 58 has been amended to recite "outputting the lower-level specification." Claim 59 has been amended for the sake of consistency with amended claim 58. Accordingly, Applicants respectfully submit that the rejection of claim 58 should be withdrawn.

Objection to the Specification

The Action objects to the specification, requesting that the cross-reference information be updated. Applicants note that the preliminary amendment dated February 13, 2004, changed the language of the priority claim. Applicants have updated the cross-reference information relative to the February 13, 2004, amendment. Applicants submit that with entry of this Amendment the cross-reference information is hereby updated. Accordingly, Applicants respectfully request that the objection to the specification be removed.

Patentability of Claims 51-62 and 72-74 over Panchul under 35 U.S.C. 102(e)

The Action rejects claims 51-62 and 72-74 under 35 U.S.C. 102(e) as being unpatentable over U.S. Patent 6,226,776 to Panchul et al. ("Panchul"). These rejections are respectfully traversed. For a 102(e) rejection to be proper, the applied art must show each and every element as set forth in a claim. (*See* MPEP § 2131.01.) However, Panchul fails to do so.

Independent Claim 51

Independent claim 51 recites:

> accepting a programming language specification, the programming language specification including plural calls to plural instances of a unit class, wherein a first call of the plural calls maps to a first instance of the plural instances of the unit class, wherein a second call of the plural calls maps to a second instance of the plural instances of the unit class, and wherein the programming language specification lacks explicit concurrency modeling for the plural instances of the unit class; and
>
> transforming the programming language specification into a lower-level specification, wherein the transforming includes generating lower-level description for handling concurrent execution of units represented by the plural instances of the unit class in the programming language specification.

Independent claim 51 is directed to accepting a programming language specification that includes multiple calls to multiple instances of a unit class. A first call of the multiple calls maps to a first instance of the multiple instances, and a second call of the multiple calls maps to a second instance of the multiple instances. The programming language specification lacks explicit concurrency modeling for the multiple instances of the unit class.

The Action relies on Panchul disclosing the above-quoted language of claim 51. Applicants respectfully disagree.

The Action cites various passages in Panchul, which are each addressed below, against the above-quoted language of claim 51. For example, at col. 14, lines 51-53, Panchul states that "various ANSI C functions can be classified 'simple expression functions,' such as 'func1' and 'func2' shown in FIG. 3A." Panchul later states that "FIG. 3A illustrates C-type programming language expressions 'a+b c<<3' and '(a b) & c,'" where "'a', 'b', and 'c' are integer variables." (Col. 14, line 66 to col. 15, line 1; *see also* col. 5, lines 14-27, col. 6, lines 6-22, col. 15, lines 31-43, col. 21, lines 28-46, Figures 14A and 14B.) Panchul also states that "FIG. 3B illustrates that these expressions are mapped into RTL Verilog expressions." (Col. 15, lines 1-3.) The description of FIGs. 3A and 3B in Panchul is understood to describe a single expression of each of two distinct functions and a mapping of the two expressions to RTL, which involves two distinct functions (not multiple calls to one) and involves "C-type functions" (not instances or a unit class). It does not teach or suggest a programming language specification including plural calls to plural instances of a unit class, let alone "accepting a programming language specification, the programming language specification including plural calls to plural instances

of a unit class, wherein a first call of the plural calls maps to a first instance of the plural instances of the unit class, wherein a second call of the plural calls maps to a second instance of the plural instances of the unit class, and wherein the programming language specification lacks explicit concurrency modeling for the plural instances of the unit class," as recited in independent claim 51.

Independent claim 51 is also directed to transforming the programming language specification into a lower-level specification. The transforming includes generating lower-level description for handling concurrent execution of units represented by the multiple instances of the unit class in the programming language specification.

The Action relies on Panchul disclosing the above-quoted language of claim 51. Applicants respectfully disagree.

The Action cites various passages in Panchul, which are each addressed below, against the above-quoted language of claim 51. For example, at col. 14, lines 42-48, Panchul describes "mapping predetermined C-type programming language expressions to functionally equivalent HDL program language expressions." Panchul indicates that "FIG. 3B represents the function shown in FIG. 3A compiled into RTL Verilog HDL." (Col. 14, lines 46-48; *see also* col. 5, lines 14-27, col. 6, lines 6-22, col. 15, lines 31-43, col. 21, lines 28-46, Figures 14A and 14B.) The mapping in Panchul is understood to describe the mapping of C-type programming language expressions to functionally equivalent HDL program language expressions, where "selected C-type functions that can execute simultaneously" "are compiled into a plurality of executable HDL program language expressions that operate in parallel." (Col. 15, lines 31-34.) This involves mapping *different* C-type functions that can execute simultaneously into HDL expressions that operate in parallel and involves C-type functions (not instances or unit classes). Thus, Panchul does not teach or suggest "transforming the programming language specification into a lower-level specification, wherein the transforming includes generating lower-level description for handling *concurrent execution of units represented by the plural instances of the unit class* in the programming language specification," as recited in independent claim 51.

Therefore, Applicants respectfully submit that Panchul does not show each and every element as set forth in claim 51. Accordingly, Applicants respectfully request that the 35 U.S.C. § 102(e) rejection be withdrawn from independent claim 51.

Dependent Claims 52-57

Claims 52-57 ultimately depend on independent claim 51. Thus, for at least the reasons set forth above with respect to independent claim 51, claims 52-57 should also be in condition for allowance. These claims are also independently patentable. Accordingly, Applicants respectfully request that the 35 U.S.C. § 102(e) rejection be withdrawn from dependent claims 52-57.

Independent Claim 58

Independent claim 58, as amended, recites:

> a design input module for accepting an algorithmic specification, the algorithmic specification including plural unit calls that map to plural different instances of a unit, thereby indicating parallel execution of the plural unit calls; and
>
> a hardware description language transformer for transforming the algorithmic specification into a lower-level specification and outputting the lower-level specification, wherein the transformer adds code into the lower-level specification for handling the parallel execution of the plural unit calls.

Independent claim 58 is directed to a design tool that includes a design input module and a hardware description language transformer. The design input module is operable to accept an algorithmic specification that includes multiple unit calls that map to multiple different instances of a unit, thereby indicating parallel execution of the multiple unit calls. The transformer is operable to transform the algorithmic specification into a lower-level specification. The transformer is also operable to add code into the lower-level specification for handling the parallel execution of the plural unit calls.

The Action relies on Panchul disclosing the above-quoted language of claim 58. Applicants respectfully disagree.

The Action cites various passages in Panchul, which are each addressed below, against the above-quoted language of claim 58. For example, at col. 13, lines 34-36, Panchul states that "the preliminary hardware design is compiled into a hardware description language (HDL) synthesizable design." This involves compilation into HDL but does not teach or suggest the above-quoted "plural unit calls that map to plural different instances of a unit" or "parallel execution" language of claim 58.

Panchul later indicates that "FIGS. 3A and 3B also illustrate a plurality of selected C-type functions that can execute simultaneously," where the "selected C-type functions that can execute simultaneously" "are compiled into a plurality of executable HDL program language expressions that operate in parallel." (Col. 15, lines 31-34; *see also* col. 5, lines 14-27, col. 6, lines 6-22, col. 14, lines 46-48, col. 15, lines 34-43, col. 21, lines 28-46, Figures 14A and 14B.) This involves mapping *different* C-type functions that can execute simultaneously into HDL expressions that operate in parallel and involves C-type functions (not instances). Thus, Panchul does not teach or suggest "plural unit calls that map to plural *different instances of a unit*, thereby indicating parallel execution of the plural unit calls" "wherein the transformer adds code into the lower-level specification for handling the parallel execution of the plural unit calls," as recited in independent claim 58.

Therefore, Applicants respectfully submit that Panchul does not show each and every element as set forth in claim 58. Accordingly, Applicants respectfully request that the 35 U.S.C. § 102(e) rejection be withdrawn from independent claim 58.

Dependent Claims 59-62

Claims 59-62 ultimately depend on independent claim 58. Thus, for at least the reasons set forth above with respect to independent claim 58, claims 59-62 should also be in condition for allowance. These claims are also independently patentable. Accordingly, Applicants respectfully request that the 35 U.S.C. § 102(e) rejection be withdrawn from dependent claims 59-62.

Independent Claim 72

Independent claim 72 recites:

> providing a programming language specification, the programming language specification including plural unit calls that map to plural different instances of a unit class, thereby indicating parallel execution of the plural unit calls; and
>
> receiving a lower-level specification produced by transforming the programming language specification into the lower-level specification, wherein the transforming includes adding code into the lower-level specification for handling the parallel execution of the plural unit calls.

Independent claim 72 is directed to providing a programming language specification that includes multiple unit calls that map to multiple different instances of a unit class, thereby indicating parallel execution of the multiple unit calls.

The Action relies on Panchul disclosing the above-quoted language of claim 72. Applicants respectfully disagree.

The Action cites various passages in Panchul, which are each addressed below, against the above-quoted language of claim 72. For example, at col. 14, lines 51-53, Panchul states that "various ANSI C functions can be classified 'simple expression functions,' such as 'func1' and 'func2' shown in FIG. 3A." Panchul later states that '"FIG. 3A illustrates C-type programming language expressions 'a+b c<<3' and '(a b) & c,' where "'a', 'b', and 'c' are integer variables." (Col. 14, line 66, to col 15, line 1; *see also* col. 5, lines 14-27, col. 6, lines 6-22, col. 15, lines 31-43, col. 21, lines 28-46, Figures 14A and 14B.) Panchul also states that "FIG. 3B illustrates that these expressions are mapped into RTL Verilog expressions." (Col. 15, lines 1-3.) The description of FIGs. 3A and 3B in Panchul is understood to describe a single expression of each of two distinct functions and a mapping of the two expressions to RTL, which involves two distinct functions (not multiple calls to one) and involves "C-type functions" (not instances or a unit class). It does not teach or suggest a programming language specification including plural calls that map to plural different instances of a unit class, let alone "providing a programming language specification, the programming language specification including plural unit calls that map to plural different instances of a unit class, thereby indicating parallel execution of the plural unit calls," as recited in independent claim 72.

Independent claim 72 is also directed to receiving a lower-level specification produced by transforming the programming language specification into the lower-level specification. Code is added into the lower-level specification for handling the parallel execution of multiple unit calls.

The Action relies on Panchul disclosing the above-quoted language of claim 72. Applicants respectfully disagree.

The Action cites various passages in Panchul, which are each addressed below, against the above-quoted language of claim 72. For example, at col. 13, lines 34-36, Panchul states that "the preliminary hardware design is compiled into a hardware description language (HDL) synthesizable design." This involves compilation into HDL but does not teach or suggest the

above-quoted "plural unit calls that map to plural different instances of a unit class" or "parallel execution" language of claim 72.

Panchul later indicates that "FIGS. 3A and 3B also illustrate a plurality of selected C-type functions that can execute simultaneously," where the "selected C-type functions that can execute simultaneously" "are compiled into a plurality of executable HDL program language expressions that operate in parallel." (Col. 15, lines 31-34; *see also* col. 5, lines 14-27, col. 6, lines 6-22, col. 14, lines 46-48, col. 15, lines 34-43, col. 21, lines 28-46, Figures 14A and 14B.) This involves mapping *different* C-type functions that can execute simultaneously into HDL expressions that operate in parallel and involves C-type functions (not instances or a unit class). Thus, Panchul does not teach or suggest "plural unit calls that map to plural *different instances of a unit class*, thereby indicating parallel execution of the plural unit calls" "wherein the transforming includes adding code into the lower-level specification for handling the parallel execution of the plural unit calls," as recited in independent claim 72.

Therefore, Applicants respectfully submit that Panchul does not show each and every element as set forth in claim 72. Accordingly, Applicants respectfully request that the 35 U.S.C. § 102(e) rejection be withdrawn from independent claim 72.

Dependent Claims 73-74

Claims 73-74 ultimately depend on independent claim 72. Thus, for at least the reasons set forth above with respect to independent claim 72, claims 73-74 should also be in condition for allowance. These claims are also independently patentable. Accordingly, Applicants respectfully request that the 35 U.S.C. § 102(e) rejection be withdrawn from dependent claims 73-74.

Double Patenting

The Action rejects claims 51, 58, and 72 under the judicially created doctrine of obviousness-type double patenting as being unpatentable over claims 9, 14, and 14, respectively, of U.S. Patent No. 6,701,501. Applicants respectfully disagree with the obviousness-type double patenting rejections and with the Examiner's characterizations of the claims of the present application and U.S. Patent No. 6,701,501. To expedite prosecution, however, Applicants submit herewith a terminal disclaimer that renders moot the obviousness-type double patenting

rejections. Applicants respectfully request that the rejections of claims 51, 58, and 72 be withdrawn.

Request for Interview

Applicants believe the application to be allowable. If any issues remain, however, the Examiner is formally requested to contact the undersigned attorney at (503) 226-7391 prior to issuance of the next communication in order to arrange a telephonic interview.

This request is being submitted under MPEP § 713.01, which indicates that an interview may be arranged in advance by a written request.

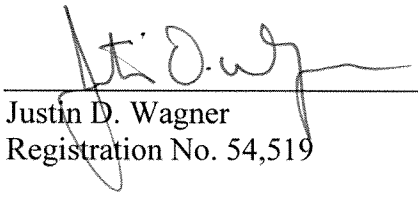
Conclusion

The claims in their present form should now be allowable. Such action is respectfully requested.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 595-5300
Facsimile: (503) 595-5301

By 

Justin D. Wagner
Registration No. 54,519